Weighted EF1 and PO Allocations with Few Types of Agents or Chores

Jugal Garg, **Aniket Murhekar**, **John Qin**
University of Illinois at Urbana-Champaign
{jugal, aniket2, johnqin2}@illinois.edu

Abstract

We investigate the existence of fair and efficient allocations of indivisible chores to asymmetric agents who have unequal entitlements or weights. We consider the fairness notion of weighted envy-freeness up to one chore (wEF1) and the efficiency notion of Pareto-optimality (PO). The existence of EF1 and PO allocations of chores to symmetric agents is a major open problem in discrete fair division, and positive results are known only for certain structured instances. In this paper, we study this problem for a more general setting of asymmetric agents and show that an allocation that is wEF1 and PO exists and can be computed in polynomial time for instances with:

- Three types of agents where agents with the same type have identical preferences but can have different weights.
- Two types of chores.

For symmetric agents, our results establish that EF1 and PO allocations exist for three types of agents and also generalize known results for three agents, two types of agents, and two types of chores. Our algorithms use a weighted picking sequence algorithm as a subroutine; we expect this idea and our analysis to be of independent interest.

1 Introduction

Fair division is a ubiquitous problem in many disciplines, such as computer science, economics, social choice, and multi-agent systems. While the formal study began with the cake-cutting problem proposed by Steinhaus [Steinhaus, 1949], which concerned the fair division of a divisible good, the fair division of indivisible items has received considerable attention in recent times (see excellent surveys [Aziz *et al.*, 2022; Amanatidis *et al.*, 2023]). Although fairness of an allocation is inherently subjective, envy-freeness (EF) is a quintessential and well-established notion of fairness [Foley, 1967]. Envy-freeness requires that every agent (weakly-)prefers the items allocated to her to those allocated to others. Unfortunately, EF allocations need not exist when items are indivisible, as can be seen from the simple example of assigning one task to two agents. Hence, relaxations of EF have been defined to qualify fairness in the discrete case, with envy-free up one item (EF1) being one such popular relaxation. When agent preferences are monotone, EF1 allocations always exist and can be computed in polynomial time [Lipton *et al.*, 2004; Bhaskar *et al.*, 2021].

A fair allocation can be quite sub-optimal in terms of overall efficiency: a set of tasks could be allocated so that every agent is assigned tasks they are bad at; as a result, agents may not envy each other by much, but the allocation is inefficient. It is, therefore, natural to seek allocations that are both fair as well as efficient. The classic notion of economic efficiency is Pareto-optimality (PO): an allocation is PO if there is no re-allocation such that every agent weakly prefers the re-allocation while some agents strictly prefer it. Allocations that are simultaneously EF1 and PO are thus highly desirable, and many works have investigated the existence and fast computation of such allocations in various settings, e.g., [Caragiannis *et al.*, 2016; Barman *et al.*, 2018; Garg and Murhekar, 2021; Ebadian *et al.*, 2022; Garg *et al.*, 2022; Garg *et al.*, 2023; Aziz *et al.*, 2019; Aziz *et al.*, 2023]. A common assumption in these works, including ours, is that agent preferences are additive, i.e., the value to an agent from a set of items is the sum of the values of items in the set. Under additive preferences, merely checking if an allocation is PO is a co-NP-hard problem. This adds to the challenge of investigating the existence and computation of EF1 and PO allocations.

Further, the difficulty of the problem is significantly influenced by the nature of the items, as is the definition of EF1. When items are goods and provide value to the agents receiving them, in an EF1 allocation, every agent prefers her bundle to that of another after the removal of one good from the other agent's bundle. For goods, an EF1 and PO allocation is known to exist and can be computed in pseudo-polynomial time. In a pivotal paper, [Caragiannis *et al.*, 2016] proved that an allocation with the highest Nash welfare — product of the agents' utilities — is both EF1 and PO. This approach does not lead to fast computation since computing an allocation with maximum Nash welfare is APX-hard. Remarkably, [Barman *et al.*, 2018] designed a pseudo-polynomial time algorithm for this problem using the idea of a competitive equilibrium. In this approach, agents are endowed with a fictitious amount of money, the goods are assigned prices, and each

Instance	EF1 + fPO	wEF1 + fPO
General Additive	?	?
Two agents	✓[Aziz *et al.*, 2019]	✓[Wu *et al.*, 2023]
Three agents	✓[Garg *et al.*, 2023]	✓Theorem 1
Two-agent-types	✓[Garg *et al.*, 2023]	✓Theorem 1
Three-agent-types	✓Theorem 1	✓Theorem 1
Two-chore-types	✓[Aziz *et al.*, 2023]	✓Theorem 2
Bivalued	✓[Garg *et al.*, 2022; Ebadian *et al.*, 2022]	✓[Wu *et al.*, 2023]

Table 1: State-of-the-art for EF1/wEF1+fPO allocation of indivisible chores. ✓ denotes existence/polynomial-time algorithm, ? denotes (non-)existence is unknown. Colored cells highlight our results.

agent is allocated goods that give them 'maximum value-for-money'. The latter ensures that the allocation is fractionally PO (fPO), an efficiency property stronger than PO. While the allocation is not EF1, they transfer goods between agents to reduce envy and make appropriate price changes to maintain PO. Using involved potential function arguments, they prove their algorithm terminates with an EF1 and PO allocation. Later, [Garg and Murhekar, 2021] showed an EF1 and fPO allocation can be computed in pseudo-polynomial time and in polynomial time when agents have a constant number of different values for the goods. Despite these results, designing a polynomial time algorithm remains an open problem.

On the other hand, when items are chores and impose a cost on agents receiving them, in an EF1 allocation every agent prefers her bundle to that of another after the removal of one chore from her bundle. For chores, even the existence of an EF1 and PO allocation is unclear, let alone computation. At first sight, it may seem that the case of goods and chores are similar, and techniques from the goods setting should be directly adaptable to chores. However, this does not seem to be the case. Firstly, a welfare function like Nash welfare guaranteeing EF1 is not known for chores. Secondly, while the competitive equilibrium approach is promising since it guarantees PO, showing that algorithms using this approach terminate has proved to be challenging. As a result, the existence and computation of an EF1 and PO allocation of chores remains a challenging open problem in discrete fair division. To understand the 'source of hardness' of this problem, a series of works have focused on identifying structured instances where the problem becomes tractable, i.e., where EF1 and PO allocations exist. These classes include instances with (i) identical agents (folklore), (ii) two agents [Aziz *et al.*, 2019], (iii) bivalued disutilities [Garg *et al.*, 2022; Ebadian *et al.*, 2022], (iv) three agents [Garg *et al.*, 2023], (v) two types of agents [Garg *et al.*, 2023], and (vi) two types of chores [Aziz *et al.*, 2023]. While the computation of EF1 and PO allocations is fairly straightforward for (i) and (ii) using ideas for goods, that for classes (iii) - (vi) is non-trivial. These results follow from carefully designed algorithms, all of which use the competitive equilibrium framework but require involved potential function arguments to prove termination. Table 1 lists these results.

All of the above works assume that agents have equal entitlements, capacities, or stakes. This is a limiting assumption, as practical scenarios may involve agents with different capacities for handling workload. For example, Alice, a teaching faculty at a university, may agree to teach twice as many courses as Bob, a research faculty. Thus, Alice will only envy Bob if she feels the workload from the courses assigned to her is more than twice that of Bob. Likewise, the dissolution of a partnership poses the problem of fairly dividing liabilities; in this context, it is only natural that they be divided according to the entitlements/shares of the partners. These scenarios motivate the definition of weighted envy-freeness (wEF) and its relaxation weighted envy-freeness up to one item (wEF1) in the discrete case. Naturally, like the unweighted case, one seeks fair and efficient allocations, which leads us to the existence and computation of a wEF1 and PO allocation of chores. This question is interesting not only because it generalizes an important open problem in discrete fair division but also because it naturally models practical chore division scenarios.

For goods, [Chakraborty *et al.*, 2020] showed that a wEF1 (without PO) allocation can be computed via a *weighted picking sequence algorithm*. This algorithm proceeds in m rounds until all m goods are allocated. In each round, a particular agent is chosen who picks her favorite good among the remaining goods. For chores, the existence of wEF1 allocations was only recently shown [Wu *et al.*, 2023] via a modification of the weighted picking sequence algorithm, and develops a novel analysis technique. [Chakraborty *et al.*, 2020] showed that wEF1 and PO allocation exists and can be computed in pseudo-polynomial time for goods by adapting the algorithm of [Barman *et al.*, 2018] to the weighted case. For chores, [Wu *et al.*, 2023] also showed that wEF1 and PO allocation can be computed for two agents and for bivalued chores (classes (ii), (iii) above). The existence of wEF1 and PO allocations for chores remained an open problem, including for the subclasses (iv)-(vi).

1.1 Our Contributions

In this work, we study the existence and computation of wEF1 and PO allocations of chores to agents with unequal entitlements. We prove that a wEF1 and fPO allocation exists and can be computed in polynomial time for instances with

- Three types of agents (Theorem 1). In a *three-agent-type* instance, the disutility function of an agent is one of three given functions.
- Two types of chores (Theorem 2). In a *two-chore-type* instance, the chores can be partitioned into two sets, each containing copies of the same chore.

Significance. Theorem 1 establishes another class for which EF1 and PO allocations exist, namely three-agent-type instances. Theorem 1 also subsumes previous works showing the existence of EF1 and PO allocations for (i) two agents [Aziz *et al.*, 2019], (ii) three agents [Garg *et al.*, 2023], and (iii) two-agent-types [Garg *et al.*, 2023]. We note that improving the existence from three agents to three types of agents in the symmetric setting is itself significant, e.g., for another popular fairness notion of *maximin share (MMS)*, MMS allocations exist for two agents but not for two types of agents [Feige *et al.*, 2021]. Theorem 2 subsumes previous work showing EF1 and PO allocations exist for two-chore-type instances [Aziz *et al.*, 2023], and also provides an alternative algorithm for the symmetric case. Moreover, our ideas can also be applied to the goods setting to show that wEF1 and PO allocations can be computed in polynomial time for these classes. We record our results in Table 1.

Our results can also be of practical significance in certain settings. For instance, a cluster may have machines with three types of processing power, e.g., CPUs, GPUs, and TPUs (three types of agents). Likewise, allocation problems may involve jobs that are either heavy or light (two types of chores).

Techniques. We first remark that simply replacing an agent with as many copies of the agent as their weight, computing an EF1 and PO allocation, and combining the allocation of all the copies does not guarantee a wEF1 and PO allocation; see our full paper [Garg *et al.*, 2024] for an example. Our algorithms use the competitive equilibrium framework to ensure fPO (hence PO). To obtain wEF1, chores are transferred from one set of agents to another while performing appropriate changes to the chore payments[1] to maintain a competitive allocation. However, these transfers and payment changes are carefully and selectively performed so that we can guarantee the termination of our algorithms.

We first design Algorithm 1 to compute a wEF1 and fPO allocation for three-agent-type instances using a novel combination of weighted picking sequence and competitive equilibrium framework. We group agents according to their type and allocate chores to agents in a group using a 'weighted picking sequence' algorithm. We begin by allocating all chores to one group and transferring chores away from this group while ensuring that agents in the other two groups do not wEF1-envy each other. For fPO, we carefully maintain the allocation at a competitive equilibrium throughout the algorithm.

We next design Algorithm 3 to compute a wEF1 and fPO allocation for two-chore-type instances. We first observe that fPO allocations in two-chore-type instances follow a certain 'ordered' structure [Aziz *et al.*, 2023]. Leveraging this idea, we initially allocate all chores to one agent called the pivot and repeatedly transfer a chore away from the pivot. These transfers respect the ordered structure ensuring fPO and are performed until we eventually obtain a wEF1 allocation, or conclude that the initial choice of the pivot is incorrect. We argue that there exists some choice of the pivot for which the algorithm results in a wEF1 and fPO allocation.

The correctness and termination of our algorithms rely on several involved and novel potential function arguments. In particular, both our results crucially utilize novel properties of a *weighted picking sequence* algorithm for chores (e.g. Lemmas 3 and 4). We believe this may be of independent interest, and may find use in algorithm design for fair division to asymmetric agents in the future.

2 Preliminaries

An instance (N, W, M, D) of the fair division problem with chores consists of a set $N = [n]$ of n agents, a list $W = \{w_i\}_{i \in N}$ with $w_i > 0$ denoting the weight of agent i, a set $M = [m]$ of m indivisible chores, and a list $D = \{d_i\}_{i \in N}$, where $d_i : 2^M \to \mathbb{R}_{\geq 0}$ is agent i's *disutility* function over the chores. We let $d_i(j)$ denote the disutility of chore j for agent i. We assume disutility functions are additive, so that for every $i \in N$ and $S \subseteq M$, $d_i(S) = \sum_{j \in S} d_i(j)$. We consider the following structured classes. An instance (N, W, M, D) is said to be a:

- *k-agent-type* instance if there are k *types* of agents. That is, there is a set $C = \{c_1, \ldots, c_k\}$ of $k \in \mathbb{N}$ disutility functions s.t. for all $i \in N$, $d_i \in C$.
- *k-chore-type* instance if there are k *types* of chores. That is, the set of chores can be partitioned as $M = \bigcup_{\ell \in [k]} M_\ell$, where each set M_ℓ consists of copies of the same chore.

An *integral allocation* $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ is a partition of the chores into n bundles, where agent i receives bundle $\mathbf{x}_i \subseteq M$ and gets disutility $d_i(\mathbf{x}_i)$. In a *fractional allocation* $\mathbf{x} \in [0,1]^{n \times m}$, chores are divisible and $x_{ij} \in [0,1]$ denotes the fraction of chore j given to agent i. Here $d_i(\mathbf{x}_i) = \sum_{j \in M} d_i(j) \cdot x_{ij}$. We will assume that allocations are integral unless explicitly stated otherwise.

Fairness and efficiency notions. An allocation $\mathbf{x}$ satisfies:

1. Envy-free up to one chore (EF1) for symmetric agents if for all $i, h \in N$, $d_i(\mathbf{x}_i \setminus j) \leq d_i(\mathbf{x}_h)$ for some $j \in \mathbf{x}_i$.
2. Weighted envy-free up to one chore (wEF1) if for all $i, h \in N$, $\frac{d_i(\mathbf{x}_i \setminus j)}{w_i} \leq \frac{d_i(\mathbf{x}_h)}{w_h}$ for some $j \in \mathbf{x}_i$.
3. Pareto-optimal if there is no allocation $\mathbf{y}$ that dominates $\mathbf{x}$. An allocation $\mathbf{y}$ dominates an allocation $\mathbf{x}$ if for all $i \in N$, $d_i(\mathbf{y}_i) \leq d_i(\mathbf{x}_i)$, and there exists $h \in N$ such that $d_h(\mathbf{y}_h) < d_h(\mathbf{x}_h)$.
4. Fractionally Pareto-optimal if there is no fractional allocation that dominates $\mathbf{x}$. An fPO allocation is clearly PO, but not vice-versa.

Competitive equilibrium of chores. In the Fisher market model for chores, we associate payments $\mathbf{p} = (p_1, \ldots, p_m)$ with the chores. Each agent i aims to earn a desired *minimum payment* of $e_i \geq 0$ by performing chores in exchange for payment. In a (fractional) allocation $\mathbf{x}$ with payments $\mathbf{p}$, the *earning* of agent i is $\mathbf{p}(\mathbf{x}_i) = \sum_{j \in M} p_j \cdot x_{ij}$. For each agent i, we define the *pain-per-buck* ratio α_{ij} of chore j as $\alpha_{ij} = d_i(j)/p_j$ and the *minimum-pain-per-buck* (MPB) ratio as $\alpha_i = \min_{j \in M} \alpha_{ij}$. Further, we let $\mathsf{MPB}_i = \{j \in M \mid d_i(j)/p_j = \alpha_i\}$ denote the set of chores which are MPB for agent i under payments $\mathbf{p}$.

[1]Chores have attached payments while goods have prices.

We say that $(\mathbf{x},\mathbf{p})$ is a *competitive equilibrium* (CE) if (i) for all $j \in M$, $\sum_{i\in N} x_{ij} = 1$, i.e., all chores are completely allocated, (ii) for all $i \in N$, $\mathbf{p}(\mathbf{x}_i) = e_i$, i.e., each agent receives her minimum payment, and (iii) for all $i \in N$, $\mathbf{x}_i \subseteq \mathsf{MPB}_i$, i.e., agents receive only chores which are MPB for them. The First Welfare Theorem [Mas-Colell *et al.*, 1995] shows that competitive equilibria are efficient, i.e., for a CE $(\mathbf{x},\mathbf{p})$, the allocation $\mathbf{x}$ is fPO.

For a CE $(\mathbf{x},\mathbf{p})$ where $\mathbf{x}$ is integral, we let $\mathbf{p}_{-1}(\mathbf{x}_i) := \min_{j\in\mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$ denote the payment agent i receives from $\mathbf{x}_i$ excluding her highest paying chore.

Definition 1. (Weighted payment EF1) An allocation $(\mathbf{x},\mathbf{p})$ is said to be *weighted payment envy-free up to one chore* (wpEF1) if for all $i,h \in N$ we have $\frac{\mathbf{p}_{-1}(\mathbf{x}_i)}{w_i} \le \frac{\mathbf{p}(\mathbf{x}_h)}{w_h}$. Agent i *wpEF1-envies* h if $\frac{\mathbf{p}_{-1}(\mathbf{x}_i)}{w_i} > \frac{\mathbf{p}(\mathbf{x}_h)}{w_h}$.

The following lemma shows a sufficient condition for computing a wEF1 and PO allocation.

Lemma 1. *Let $(\mathbf{x},\mathbf{p})$ be a CE with $\mathbf{x}$ integral. If $(\mathbf{x},\mathbf{p})$ is wpEF1, then $\mathbf{x}$ is wEF1 and fPO.*

Proof. Let α_i be the MPB ratio of agent i in $(\mathbf{x},\mathbf{p})$. Consider any pair of agents $i,h \in N$. We have:

$$\min_{j\in\mathbf{x}_i} \frac{d_i(\mathbf{x}_i \setminus j)}{w_i} = \frac{\alpha_i \cdot \mathbf{p}_{-1}(\mathbf{x}_i)}{w_i} \le \frac{\alpha_i \cdot \mathbf{p}(\mathbf{x}_h)}{w_h} \le \frac{d_i(\mathbf{x}_h)}{w_h},$$

where the first and last transitions use that $(\mathbf{x},\mathbf{p})$ is on MPB, and the middle inequality uses Definition 1. This shows that $\mathbf{x}$ is wEF1. Moreover, the First Welfare Theorem [Mas-Colell *et al.*, 1995] implies that the allocation $\mathbf{x}$ is fPO since $(\mathbf{x},\mathbf{p})$ is a CE. □

An agent ℓ is called a *weighted least earner* (wLE) among agent set A if $\ell \in \mathsf{argmin}_{i\in A} \frac{\mathbf{p}(\mathbf{x}_i)}{w_i}$. An agent b is called a *weighted big earner* (wBE) among agent set A if $b \in \mathsf{argmax}_{i\in A} \frac{\mathbf{p}_{-1}(\mathbf{x}_i)}{w_i}$. The next lemma shows the importance of the wBE and wLE agents.

Lemma 2. *An integral CE $(\mathbf{x},\mathbf{p})$ is wpEF1 if and only if a wBE b does not wpEF1-envy a wLE ℓ.*

Proof. ($\Rightarrow$) If $(\mathbf{x},\mathbf{p})$ is wpEF1 then clearly b does not wpEF1-envy ℓ.
($\Leftarrow$) Suppose b does not wpEF1-envy ℓ. Then the following shows that no agent i wpEF1-envies any agent h, implying that $(\mathbf{x},\mathbf{p})$ must be wpEF1.

$$\frac{\mathbf{p}_{-1}(\mathbf{x}_i)}{w_i} \le \frac{\mathbf{p}_{-1}(\mathbf{x}_b)}{w_b} \le \frac{\mathbf{p}(\mathbf{x}_\ell)}{w_\ell} \le \frac{\mathbf{p}(\mathbf{x}_h)}{w_h},$$

where the first and last inequalities use the definitions of wBE and wLE, and the middle inequality uses that b does not wpEF1-envy ℓ. □

3 wEF1 and fPO Allocations in Three-Agent-Types Instances

We now study the existence and computation of wEF1 and fPO allocations in instance with three agent types. We devise a polynomial time algorithm, Algorithm 2, and show that it computes a wEF1 and fPO allocation for such instances. We therefore have the following theorem.

Algorithm 1 Weighted Picking Sequence (WPS) Algorithm

Input: Agents N with identical disutility function $d(\cdot)$, set of chores M s.t. $d_1 \ge \cdots \ge d_m$
Output: An wEF1 allocation $\mathbf{x}$

```
1: For each $i \in N$, $\mathbf{x}_i \leftarrow \emptyset$, $s_i \leftarrow 0$
2: for $j = 1$ to $m$ do
3:     $i \leftarrow \arg\min_{k\in N} \frac{s_k}{w_k}$; ties broken in favour of smaller index agent
4:     $\mathbf{x}_i \leftarrow \mathbf{x}_i \cup \{j\}$, $s_i \leftarrow s_i + 1$
5: return $\mathbf{x}$
```

Theorem 1. *In any three-agent-types instance, a wEF1 and fPO allocation exists, and can be computed in polynomial time.*

We remark that our result generalizes the following previously known results regarding the polynomial time computability of allocations that are: (i) EF1+fPO for three unweighted agents [Garg *et al.*, 2023], (ii) EF1+fPO for two types of unweighted agents [Garg *et al.*, 2023], (iii) wEF1+fPO for two agents [Wu *et al.*, 2023].

Let $N = N_1 \sqcup N_2 \sqcup N_3$ be a partition of the set of agents into three sets, called agent *groups*, each containing agents of the same type. Let $d_i(\cdot)$ be the disutility function of agents in group N_i, for $i \in \{1,2,3\}$. Note that agents in the same group can have different entitlements.

3.1 Algorithm Description

Through the execution of Algorithm 2 we let M_i denote the set of chores allocated to group N_i. Initially, all chores are allocated to a single group, N_1, i.e., $M_1 = M$, and $M_2 = M_3 = \emptyset$. At each point, the set of chores M_i is allocated to the group N_i by using the *weighted picking sequence* (WPS) procedure, Algorithm 1, described below.

Weighted Picking Sequence (WPS) Algorithm. The WPS algorithm (Algorithm 1) takes as input a set of agents N of the same type, i.e., with *identical* disutility function $d(\cdot)$, and a set of chores M. The algorithm first sorts and relabels the m chores in non-increasing order of disutility, i.e., $d_1 \ge d_2 \ge \cdots \ge d_m$. The algorithm then performs m iterations, allocating one chore in each iteration. In iteration j, chore j is allocated to an agent with the least value of $\frac{s_i}{w_i}$, where s_i denotes the number of chores allocated to agent i so far. It is known that the WPS procedure returns a wEF1 allocation for identical agents [Chakraborty *et al.*, 2020; Wu *et al.*, 2023]. Recently, [Wu *et al.*, 2023] showed that by allocating the chores in reverse order with each agent picking their least disutility chore in their turn results in a wEF1 allocation, even for non-identical agents. Since our algorithm uses the WPS algorithm to assign chores to agents of the same type, we do not require this generalization. Using the WPS algorithm to allocate chores M_i to group N_i ensures that agents within a group do not ever wEF1-envy each other, and any wEF1-envy is between agents belonging to different groups. To reduce wEF1-envy across groups, we transfer chores from one group to another. After each chore transfer the set of chores M_i gets updated and is re-allocated to the group N_i by using the WPS algorithm. We denote by

$\mathsf{WPS}(N_1, M_1) \cup \mathsf{WPS}(N_2, M_2) \cup \mathsf{WPS}(N_3, M_3)$ the allocation resulting from allocating M_i to N_i using WPS for each $i \in [3]$.

Chore transfers and Payment drops: High-level Ideas. To ensure that the resulting allocation is fPO, we attach payments to the chores and maintain that all allocations in the run of Algorithm 2 are on MPB, i.e., are competitive. Algorithm 2 performs two kinds of *steps*: (i) *chore transfers* and (ii) *payment drops*. Chore transfers involve the transfer of a chore from one group to another, and payment drops involve decreasing the payments of all chores belonging to one or two agent groups.

Initially we set the payment of chore j as $p_j = d_1(j)$. At some point in the algorithm let the *weighted big earner group* N_β be the group containing the weighted big earner (wBE) at the time. Likewise, let the *weighted least earner group* N_λ be the group containing the weighted least earner (wLE) at the time. We will call the group N_μ which contains neither as the *weighted middle earner (wME) group*. If $\beta = \lambda$ then the allocation must already be wpEF1 by Lemma 2. Initially $N_\beta = N_1$. We maintain that for the majority of the algorithm $N_\beta = N_1$, i.e., the wBE is in N_1, and transfer chores unilaterally from N_1 to N_2 and N_3. We also aim to maintain that agents in N_2 and N_3 do not wpEF1-envy each other, and perform chore transfers between N_2 and N_3 to eliminate any arising wpEF1-envy.

Since N_1 only loses chores, in at most m chore transfers from N_1 it must be that either the allocation becomes wpEF1 (hence wEF1), or the wBE ceases to be in N_1. In the latter case we show that in one additional chore transfer step the allocation is wEF1. To ensure fPO and facilitate chore transfers, we perform payment drops of chores when appropriate, and maintain the MPB condition during chore transfers. That is, a chore j is transferred from group N_i to N_h only if j belongs to the MPB set MPB_h of agents in N_h, i.e., $j \in M_i \cap \mathsf{MPB}_h$.

Chore transfers and Payment drops: Details. We perform payment drops and chore transfers across groups in a careful and specific manner, as described below.

(Lines 12-14) Since we desire a wpEF1 allocation if possible we first check if there exists a chore $j \in M_\beta \cap \mathsf{MPB}_\lambda$ which can be transferred directly from the wBE group to the wLE group. If so we make this transfer.

(Lines 15-18) If no chore can be transferred from N_β to N_λ, we check if a chore in $M_\mu \cap \mathsf{MPB}_\lambda$ can be potentially transferred from N_μ to N_λ. If there is a chore $j \in M_\mu \cap \mathsf{MPB}_\lambda$ s.t. after losing j the least weighted earning of an agent in N_μ is strictly larger than the weighted earning of the current wLE, then we transfer j from N_μ to N_λ (Line 17-18). Line 16 performs this check. If not, an N_μ to N_λ transfer is *not allowed*.

(Lines 19-21) If an N_μ to N_λ transfer is not allowed, we check if an N_β to N_μ transfer is possible, i.e., there is a chore $j' \in M_\beta \cap \mathsf{MPB}_\mu$. If so, we make such a transfer.

(Lines 22-25) Otherwise, no chore can be transferred from N_β to either N_μ or N_λ. In this case we lower the payments of chores in M_μ and M_λ until a chore in M_β becomes MPB for agents in N_μ or N_λ.

(Lines 26-29) Finally if there is no chore that can be transferred from N_β or N_μ to N_λ, we lower the payments of chores in M_λ until a chore in M_β or M_μ becomes MPB for agents in N_λ.

Algorithm 2 wEF1+fPO for three agent types instances

```
Input: Three agent types instance $(N, W, M, D)$
Output: A wEF1 and fPO integral allocation $\mathbf{x}$
1: Let $N = N_1 \sqcup N_2 \sqcup N_3$ be the partition of the agents according to their type
2: Let $d_i(\cdot)$ denote the disutility function of agents in group $N_i$, for $i \in \{1, 2, 3\}$
3: $M_1 \leftarrow M, M_2 \leftarrow \emptyset, M_3 \leftarrow \emptyset$
4: $\mathbf{x} \leftarrow \mathsf{WPS}(N_1, M_1) \cup \mathsf{WPS}(N_2, M_2) \cup \mathsf{WPS}(N_3, M_3)$
5: For each $j \in M$, set $p_j \leftarrow d_1(j)$
6: while $\mathbf{x}$ is not wpEF1 do
7:     $b \leftarrow \mathsf{argmax}_{k \in N} \mathbf{p}_{-1}(\mathbf{x}_k)/w_k$   ▷ Weighted big earner
8:     $\ell \leftarrow \mathsf{argmin}_{k \in N} \mathbf{p}(\mathbf{x}_k)/w_k$   ▷ Weighted least earner
9:     $\beta \leftarrow i \in [3]$ s.t. $b \in N_i$,   ▷ Index of wBE group
10:    $\lambda \leftarrow i \in [3]$ s.t. $\ell \in N_i$   ▷ Index of wLE group
11:    $\mu \leftarrow i \in [3] \setminus \{\beta, \lambda\}$   ▷ Index of wME group
       ▷ Check for $N_\beta$ to $N_\lambda$ chore transfer
12:    if $\exists j \in M_\beta \cap \mathsf{MPB}_\lambda$ then
13:        $M_\beta \leftarrow M_\beta \setminus j, M_\lambda \leftarrow M_\lambda \cup j$
14:        $\mathbf{x} \leftarrow \bigcup_{i \in [3]} \mathsf{WPS}(N_i, M_i)$
       ▷ Check for potential $N_\mu$ to $N_\lambda$ chore transfer
15:    else if $|M_\mu \cap \mathsf{MPB}_\lambda| > 0$ then
           ▷ Check if $N_\mu$ to $N_\lambda$ chore transfer is allowed
16:        if $\exists j \in M_\mu \cap \mathsf{MPB}_\lambda$ s.t. $\min_{k \in N_\mu} \frac{\mathbf{p}(\mathbf{y}_k)}{w_k} > \frac{\mathbf{p}(\mathbf{x}_\ell)}{w_\ell}$ for $\mathbf{y} = \mathsf{WPS}(N_\mu, M_\mu \setminus j)$ then
17:            $M_\mu \leftarrow M_\mu \setminus j, M_\lambda \leftarrow M_\lambda \cup j$
18:            $\mathbf{x} \leftarrow \bigcup_{i \in [3]} \mathsf{WPS}(N_i, M_i)$
           ▷ $N_\beta$ to $N_\mu$ chore transfer
19:        else if $\exists j' \in M_\beta \cap \mathsf{MPB}_\mu$ then
20:            $M_\beta \leftarrow M_\beta \setminus j', M_\mu \leftarrow M_\mu \cup j'$
21:            $\mathbf{x} \leftarrow \bigcup_{i \in [3]} \mathsf{WPS}(N_i, M_i)$
22:        else   ▷ No chore can be transferred from $N_\beta$
               ▷ Lower payments of chores in $M_\mu \cup M_\lambda$
23:            $\gamma \leftarrow \max_{i \in \{\lambda, \mu\}, j \in M_\beta} \frac{\alpha_i}{d_i(j)/p_j}$; $\alpha_i$ is the MPB ratio of an agent in group $N_i$
24:            for $j \in M_\mu \cup M_\lambda$ do
25:                $p_j \leftarrow \gamma \cdot p_j$
26:    else   ▷ No chore can be transferred from $N_\beta$ or $N_\mu$
           ▷ Lower payments of chores in $M_\lambda$
27:        $\gamma \leftarrow \max_{j \in M_\beta \cup M_\mu} \frac{\alpha_\ell}{d_\ell(j)/p_j}$
28:        for $j \in M_\lambda$ do
29:            $p_j \leftarrow \gamma \cdot p_j$
30: return $\mathbf{x}$
```

3.2 Analysis of Algorithm 2: Overview

We begin the analysis of Algorithm 2 by proving some important properties of the WPS algorithm (Alg. 1). The following lemmas compare the disutilities of agents before and after a chore transfer.

Lemma 3. *Let N be a set of agents with the same disutility function $\mathbf{p}(\cdot)$ and M be a set of chores. Let $\mathbf{x} = \mathsf{WPS}(N, M)$ be an allocation of M to N using the WPS algorithm, and $\mathbf{y} = \mathsf{WPS}(N, M \setminus j)$, for a chore $j \in M$. Then we have:*

(i) For all $i \in N$, $\mathbf{p}(\mathbf{y}_i) \leq \mathbf{p}(\mathbf{x}_i)$.

(ii) For all $i \in N$, $\mathbf{p}_{-1}(\mathbf{y}_i) \leq \mathbf{p}_{-1}(\mathbf{x}_i)$.

(iii) For all $i, h \in N$, $\frac{\mathbf{p}(\mathbf{y}_i)}{w_i} \geq \frac{\mathbf{p}_{-1}(\mathbf{x}_h)}{w_h}$. In particular, $\frac{\mathbf{p}(\mathbf{y}_\ell)}{w_\ell} \geq \frac{\mathbf{p}_{-1}(\mathbf{x}_b)}{w_b}$, where agent ℓ has the least weighted disutility in $\mathbf{y}$ and agent b has the biggest weighted disutility up to one chore in $\mathbf{x}$.

Lemma 4. *Let N be a set of agents with the same disutility function $\mathbf{p}(\cdot)$ and M be a set of chores. Let $\mathbf{x} = \mathsf{WPS}(N, M)$ be an allocation of M to N using the WPS algorithm, and $\mathbf{y} = \mathsf{WPS}(N, M \cup j)$, for a chore $j \notin M$. Then we have:*

(i) For all $i \in N$, $\mathbf{p}(\mathbf{y}_i) \geq \mathbf{p}(\mathbf{x}_i)$.

(ii) For all $i \in N$, $\mathbf{p}_{-1}(\mathbf{y}_i) \geq \mathbf{p}_{-1}(\mathbf{x}_i)$.

(iii) For all $i, h \in N$, $\frac{\mathbf{p}_{-1}(\mathbf{y}_h)}{w_h} \leq \frac{\mathbf{p}(\mathbf{x}_i)}{w_i}$. In particular, $\frac{\mathbf{p}_{-1}(\mathbf{y}_b)}{w_b} \leq \frac{\mathbf{p}(\mathbf{x}_\ell)}{w_\ell}$, where agent b has the biggest weighted disutility up to one chore in $\mathbf{y}$ and agent ℓ has the least weighted disutility in $\mathbf{x}$.

When a set of chores M with associated payments $\mathbf{p}$ is on MPB for a group N containing agents of the same type, the disutility of a chore is proportional to its payment. Therefore, the above lemmas also hold when $\mathbf{p}$ represents the payment vector (which is proportional to the disutility vector) and $\mathbf{p}(\mathbf{x}_i)$ represents the earning of agent i in $(\mathbf{x}, \mathbf{p})$ (which is proportional to the disutility of i in $\mathbf{x}$). We next show:

Lemma 5. *Throughout the execution of Algorithm 2, no chore transfer decreases the weighted earning of the wLE.*

The following two lemmas analyze steps of Algorithm 2 which cause a group N_λ to cease being the wLE group.

Lemma 6. *If a step of Algorithm 2 results in the weighted least earner group becoming the weighted big earner group, then the resulting allocation must be wpEF1.*

Lemma 7. *If a step of Algorithm 2 results in the weighted least earner group N_λ becoming the weighted middle earning group, then agents in N_λ do not wpEF1-envy agents in the new weighted least earner group in the resulting allocation.*

To summarize, if a group N_λ ceases to be the wLE group, then either resulting allocation is wpEF1 or N_λ becomes the wME group and agents in N_λ do not wpEF1-envy agents in the new wLE group. These observations are important for proving the lemmas that follow.

We next consider steps of Algorithm 2 which cause a group N_β to cease being the wBE group. We first show:

Lemma 8. *If a step of Algorithm 2 results in the weighted big earner group becoming the weighted least earner group, then the resulting allocation must be wpEF1.*

Recall that we initially assigned all the chores to group N_1. Hence $N_\beta = N_1$ was the initial wBE group. We prove that this is the case almost throughout the execution of the algorithm. To this end, we show that N_β (i.e. N_1) loses a chore in every $\mathsf{poly}(m)$-many steps.

Lemma 9. *While the weighted big earner belongs to the group N_β and the allocation is not wpEF1, N_β must lose a chore in $\mathsf{poly}(m)$ steps.*

Note that N_1 has m chores to begin with (i.e. always $|M_1| \leq m$), and Algorithm 2 never transfers a chore to N_1. Lemma 9 therefore implies that in $\mathsf{poly}(m)$ steps either the allocation is wpEF1 or N_1 ceases to be the weighted big earner group. In the latter scenario, we show that we arrive at a wEF1 allocation in at most one more chore transfer step.

Lemma 10. *After N_1 stops being the wBE group for the last time, Algorithm 2 terminates with a wEF1 and fPO allocation after performing at most one subsequent chore transfer.*

The above discussion leads us to conclude that Algorithm 2 computes a wEF1 and fPO allocation in polynomial time for three agent types instances. This proves Theorem 1. Due to space constraints, we defer missing proofs to the full version of our paper [Garg *et al.*, 2024].

4 wEF1+fPO for Two-Chore-Type Instances

We now turn to the problem of existence and computation of a wEF1+fPO allocation for instances with two types of chores. We answer this question positively in this section. Thus, our result generalizes that of [Aziz *et al.*, 2023], who showed that EF1+fPO allocations can be computed in polynomial time for two chore type instances with unweighted agents.

Theorem 2. *In any two chore type instance, a wEF1 and fPO allocation exists and can be computed in polynomial time.*

To prove Theorem 2, we present a polynomial time algorithm, Algorithm 3, that computes a wEF1 and fPO allocation for a given two chore type instance. We note that Algorithm 3 is also an alternative algorithm to that of [Aziz *et al.*, 2023] for computing an EF1+PO allocation.

Let $M = A \sqcup B$ be a partition of the chores into two sets, each containing chores of the same type. For $X \in \{A, B\}$, we refer to a chore in set X as an X-chore, and denote by d_{iX} the disutility an agent i has for any chore in set X.

4.1 Algorithm Description

We first sort and re-label the agents in non-decreasing order of d_{iA}/d_{iB}, i.e., for $i < j$, $\frac{d_{iA}}{d_{iB}} \leq \frac{d_{jA}}{d_{jB}}$. Roughly speaking, agents with a smaller index prefer to do A-chores over B-chores, and vice-versa.

Our algorithm proceeds in at most n phases. Starting from $i = 1$, in Phase i we select agent i as the *pivot* agent. The pivot agent i is initially assigned all the chores (making i the only agent with wpEF1-envy and the unique wBE), and the payments of the chores are set according to the disutilities of the pivot. In other words, we set the payment of a chore in set X as $p_X = d_{iX}$ for $X \in \{A, B\}$. While the allocation is not wpEF1, we attempt to reduce the wpEF1-envy of i by transferring a chore from i to a wLE agent (this maintains the property that while the allocation is not wpEF1, i is the only agent with wpEF1-envy and is the unique wBE). Let ℓ_A be

Algorithm 3 wEF1+fPO for two chore types instances

```
Input: Two chore type instance $(N, W, M, D)$
Output: An integral wEF1 and fPO allocation $\mathbf{x}$
 1: Let $M = A \sqcup B$ be a partition of chores according to type
 2: for $i = 1$ to $n$ do
 3:     $\mathbf{x}_i \leftarrow M$, $\mathbf{x}_h \leftarrow \emptyset$ for $h \neq i$
 4:     Set payments as $p_A = d_{iA}$ and $p_B = d_{iB}$
 5:     while $(\mathbf{x}, \mathbf{p})$ is not wpEF1 do
            ▷ $L$ is the set of global wLE agents
 6:         $L \leftarrow \{k : k \in \arg\min_{h\in[n]} \frac{\mathbf{p}(\mathbf{x}_h)}{w_h}\}$
            ▷ Set $\ell_A$ as max. index wLE in $[i-1]$, $\max\emptyset = \infty$
 7:         $\ell_A \leftarrow \max(L \cap [i-1])$
            ▷ Set $\ell_B$ as min. index wLE in $[n]\setminus[i]$, $\min\emptyset = 0$
 8:         $\ell_B \leftarrow \min(L \cap [n] \setminus [i])$
 9:         if $\ell_A < i$ and $\mathbf{x}_i \cap A \neq \emptyset$ then
                ▷ Transfer $A$-chore from $i$ to $\ell_A$
10:             Let $a \in \mathbf{x}_i \cap A$ be an $A$-chore in $\mathbf{x}_i$
11:             $\mathbf{x}_i \leftarrow \mathbf{x}_i \setminus a$, $\mathbf{x}_{\ell_A} \leftarrow \mathbf{x}_{\ell_A} \cup a$
12:         else if $\ell_B > i$ and $\mathbf{x}_i \cap B \neq \emptyset$ then
                ▷ Transfer $B$-chore from $i$ to $\ell_B$
13:             Let $b \in \mathbf{x}_i \cap B$ be a $B$-chore in $\mathbf{x}_i$
14:             $\mathbf{x}_i \leftarrow \mathbf{x}_i \setminus b$, $\mathbf{x}_{\ell_B} \leftarrow \mathbf{x}_{\ell_B} \cup b$
15:         else: Break                          ▷ Failure in Phase $i$
16:     if $(\mathbf{x}, \mathbf{p})$ is wpEF1 then
17:         return $\mathbf{x}$
```

the index of a wLE agent among agents 1 through $(i-1)$ if such an agent exists (otherwise let $\ell_A = \infty$), with ties broken in favour of larger index (Line 7). Similarly, let ℓ_B be the index of a wLE agent among agents $(i+1)$ through n if such an agent exists (otherwise let $\ell_B = 0$), with ties broken in favour of smaller index (Line 8). We attempt to make a chore transfer as follows:

(Lines 9-11) If $\ell_A < i$ and $\mathbf{x}_i \cap A \neq \emptyset$, then there is a wLE agent to the left of the pivot agent i and i has an A-chore which is on MPB for ℓ_A. Thus, we transfer this A-chore from i to ℓ_A.

(Lines 12-14) If $\ell_B > i$ and $\mathbf{x}_i \cap B \neq \emptyset$, then there is a wLE agent to the right of pivot agent i and i has a B-chore which is on MPB for ℓ_B. Thus, we transfer this B-chore from i to ℓ_B.

(Line 15) If neither of the above cases are met, then we have encountered a failure in Phase i. It cannot be that a wLE exists on both sides of i, as then i must have been able to transfer some chore to a wLE agent (since we maintain that i is the wBE her bundle must be non-empty). This implies that either (i) a wLE agent exists only on the left side of i but i has no A-chores to transfer, or (ii) a wLE agent exists only on the right side of i but i has no B-chores to transfer. In the case of (i), we say that agent i faces an A-fail in Phase i, and in the case of (ii) we say that agent i faces a B-fail in Phase i. After facing a failure, no more transfers are done in Phase i.

Algorithm 3 terminates either when a wpEF1 allocation is found, or when all phases are completed.

4.2 Analysis of Algorithm 3: Overview

We now provide an overview of the proof of Theorem 2, by arguing that Algorithm 3 finds a wpEF1 and fPO allocation. We first show:

Lemma 11. *Throughout the run of Algorithm 3, every allocation is fPO.*

We then argue that Algorithm 3 eventually finds a wpEF1 allocation. To this end, we first show:

Lemma 12. *Throughout the execution of Phase i, if the allocation is not wpEF1 then agent i is the only weighted big earner.*

The above lemma shows that if we did not find a wpEF1 allocation in Phase i, it must be because the only wBE i could not transfer a chore to a wLE agent. That is, Phase i was terminated due to i facing either an A-fail or a B-fail. We next prove that:

Lemma 13. *If agent i faces a B-fail, then agent $(i+1)$ cannot face an A-fail.*

By definition of a failure, agent 1 cannot face an A-fail. Thus if agent 1 faces a failure, it must be due to a B-fail. By Lemma 13, we know that agent 2 cannot face an A-fail. Proceeding inductively, we conclude for each $i \in [n]$ that either agent i faces a B-fail or does not face a failure at all. However, agent n cannot face a B-fail by definition of failure. Thus it must be that there is some pivot agent $i^* \in [n]$ who did not face a failure.

Therefore, by invoking Lemma 12 we conclude that in Phase i^*, it was always possible to transfer chores from the *only* wBE agent i^* to a wLE agent, until the allocation became wpEF1. Thus, a wEF1 and fPO allocation will be found in Phase i^*. Since there are at most n phases and there are at most m chore transfers in each phase, Algorithm 3 terminates in polynomial time. This proves Theorem 2.

To prove the crucial Lemma 13, we relate the allocation returned by Algorithm 3 in Phase i with the allocation returned by running the weighted picking sequence algorithm for assigning A-chores to $[i]$ and B-chores to $[n]\setminus[i]$. Due to space constraints, we defer missing proofs to the full version of our paper [Garg *et al.*, 2024].

5 Discussion

In this work we studied the problem of computing a wEF1 and fPO allocation of chores for agents with unequal weights or entitlements. We showed positive algorithmic results for this problem for instances with three types of agents, or two types of chores. The existence of EF1 and fPO allocations of chores in the symmetric case remains a hard open problem. Our results further our understanding of this problem by contributing to the body of positive non-trivial results. Together with the result of [Wu *et al.*, 2023] concerning bivalued chores, our paper shows that wEF1 and fPO allocations exists for every structured instance known so far that admits an EF1 and fPO allocation. Our idea of combining the competitive equilibrium framework with envy-resolving algorithms like the weighted picking sequence algorithms could be an important tool in settling the problem in its full generality.

References

[Amanatidis *et al.*, 2023] Georgios Amanatidis, Haris Aziz, Georgios Birmpas, Aris Filos-Ratsikas, Bo Li, Hervé Moulin, Alexandros A. Voudouris, and Xiaowei Wu. Fair division of indivisible goods: Recent progress and open questions. *Artificial Intelligence*, 322:103965, 2023.

[Aziz *et al.*, 2019] Haris Aziz, Ioannis Caragiannis, Ayumi Igarashi, and Toby Walsh. Fair allocation of indivisible goods and chores. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, page 53–59, 2019.

[Aziz *et al.*, 2022] Haris Aziz, Bo Li, Herve Moulin, and Xiaowei Wu. Algorithmic fair allocation of indivisible items: A survey and new questions, 2022.

[Aziz *et al.*, 2023] Haris Aziz, Jeremy Lindsay, Angus Ritossa, and Mashbat Suzuki. Fair allocation of two types of chores. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '23, page 143–151, Richland, SC, 2023. International Foundation for Autonomous Agents and Multiagent Systems.

[Barman *et al.*, 2018] Siddharth Barman, Sanath Kumar Krishnamurthy, and Rohit Vaish. Finding fair and efficient allocations. In *Proceedings of the 19th ACM Conference on Economics and Computation (EC)*, pages 557–574, 2018.

[Bhaskar *et al.*, 2021] Umang Bhaskar, A. R. Sricharan, and Rohit Vaish. On Approximate Envy-Freeness for Indivisible Chores and Mixed Resources. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021)*, volume 207 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:23, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[Caragiannis *et al.*, 2016] Ioannis Caragiannis, David Kurokawa, Hervé Moulin, Ariel D. Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum Nash welfare. In *Proceedings of the 17th ACM Conference on Economics and Computation (EC)*, page 305–322, 2016.

[Chakraborty *et al.*, 2020] Mithun Chakraborty, Ayumi Igarashi, Warut Suksompong, and Yair Zick. Weighted envy-freeness in indivisible item allocation. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, page 231–239, 2020.

[Ebadian *et al.*, 2022] Soroush Ebadian, Dominik Peters, and Nisarg Shah. How to fairly allocate easy and difficult chores. In *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2022.

[Feige *et al.*, 2021] Uriel Feige, Ariel Sapir, and Laliv Tauber. A tight negative example for MMS fair allocations. In *Web and Internet Economics - 17th International Conference, WINE*, volume 13112, pages 355–372, 2021.

[Foley, 1967] Duncan Foley. Resource allocation and the public sector. *Yale Economic Essays*, 7(1):45–98, 1967.

[Garg and Murhekar, 2021] Jugal Garg and Aniket Murhekar. On fair and efficient allocations of indivisible goods. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[Garg *et al.*, 2022] Jugal Garg, Aniket Murhekar, and John Qin. Fair and efficient allocations of chores under bivalued preferences. *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, pages 5043–5050, 2022.

[Garg *et al.*, 2023] Jugal Garg, Aniket Murhekar, and John Qin. New algorithms for the fair and efficient allocation of indivisible chores. In Edith Elkind, editor, *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 2710–2718. International Joint Conferences on Artificial Intelligence Organization, 8 2023. Main Track.

[Garg *et al.*, 2024] Jugal Garg, Aniket Murhekar, and John Qin. Weighted EF1 and PO allocations with few types of agents or chores, 2024.

[Lipton *et al.*, 2004] Richard Lipton, Evangelos Markakis, Elchanan Mossel, and Amin Saberi. On approximately fair allocations of indivisible goods. In *In ACM Conference on Electronic Commerce (EC*, pages 125–131, 2004.

[Mas-Colell *et al.*, 1995] Andreu Mas-Colell, Michael D. Whinston, and Jerry R. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[Steinhaus, 1949] Hugo Steinhaus. Sur la division pragmatique. *Econometrica*, 17(1):315–319, 1949.

[Wu *et al.*, 2023] Xiaowei Wu, Cong Zhang, and Shengwei Zhou. Weighted ef1 allocations for indivisible chores. In *Proceedings of the 24th ACM Conference on Economics and Computation*, EC '23, page 1155, New York, NY, USA, 2023. Association for Computing Machinery.